

Praktikum angewandte Systemsoftwaretechnik

Aufgabe 3

Benjamin Oechslein, Daniel Lohmann, Jens Schedel, Michael Gernoth,
Moritz Strübe, Reinhard Tartler, Timo Hönig

Lehrstuhl Informatik 4

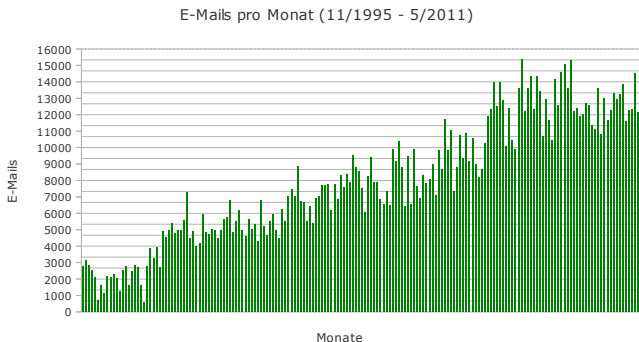
Mai, 2011

Linux-Upstream-Entwicklung

- „*The Cathedral and the Bazaar*” (Eric S. Raymond)
 - Essay über Methoden der Software-Entwicklung
 - Basiert auf Beobachtungen des Entwicklungsprozesses des Linux-Kern
- Hauptaussagen
 - „Every good work of software starts by scratching a developer’s personal itch.”
Kapitel 2: „*The Mail Must Get Through*”
 - „Release early. Release often”
Kapitel 4: „*Release Early, Release Often*”
 - „If you treat your beta-testers as if they’re your most valuable resource, they will respond by becoming your most valuable resource.”
Kapitel 5: „*Is A Rose Not A Rose?*”
 - „A security system is only as secure as its secret. Beware of pseudo-secrets.”
Kapitel 9: „*A Few More Lessons from Fetchmail*”

Linux-Upstream-Entwicklung (Fortsetzung)

- Gesamte **relevante** Kommunikation ausschließlich via E-Mail
- Wichtigster Kommunikationskanal: Linux-Kernel-Mailing-List (LKML)



- Derzeit ca. 6000 Abonnenten, Verbreitung auch über RSS-Feeds u.ä.

Programmcode-Integration

- Source-Code des Linux-Kern
 - Jede Veränderung am Linux-Kern durchläuft den gleichen Prozess
 - Egal ob Fehlerbehebung oder neue Funktionalität
- Kreuzgutachten („*Peer-Review*“)
 - Gewachsene Hierarchie mit „*Benevolent Dictator For Life*“
 - *Meistens* siegt der überlegene Ansatz
- Einflussreichste Akteure



Linus Torvalds



Andrew Morton



Alan Cox



Greg Kroah-Hartman

Kernel Janitors - Die Hausmeisterei des Linux-Kerns

- Im wesentlichen eine Gruppe von Kernel Entwicklern die kleinere, weniger gravierende Patches entgegen nimmt, sammelt, bewertet und gebündelt in den Entwicklungsprozess einbindet.
- Unterprojekt von <http://kernelnewbies.org>
- Hervorragende Anlaufstelle für Neulinge im Linux Entwicklerumfeld
- Homepage: <http://kernelnewbies.org/KernelJanitors/>

Einsenden von Entwicklungen in den Linux-Kern

Nützliche Hinweise zum Prozess in Documentation/SubmittingPatches

- Patches im *unified diff* Format
- Sinnvolle und nachvollziehbare Beschreibungen der Änderungen
- Nur **eine** logische Änderung pro Patch
- Richtige(n) Adressat(en) finden
- Keine Dateianhänge, kein MIME, kein HTML
- Basisversion genau angeben
- Knappe (70-75 Zeichen!) Kurzbeschreibung, darunter dann ausführliche Beschreibung des Patches und dessen Intention.
- Unnötige(?) Diskussionen über Geschmack (etc.) erkennen, richtig begegnen und vermeiden (↪ siehe auch: <http://bikeshed.org>)

Auffinden von zuständigen Betreuern

- Änderungen an Dateien im Linux-Kern sind immer an den jeweiligen Betreuer zur Begutachtung einzusenden
- Aufgrund der schieren Größe ist die Zuständigkeit der jeweiligen Dateien aufgeteilt
- Hilfsmittel: `scripts/get_maintainer.pl`:

```
$ scripts/get_maintainer.pl -f fs/btrfs/volumes.c
scripts/get_maintainer.pl -f fs/btrfs/volumes.c
Chris Mason <chris.mason@oracle.com>
linux-btrfs@vger.kernel.org
linux-kernel@vger.kernel.org
```

- Kann auch auf einen Patch direkt angewendet werden, um direkt die betroffenen Dateien und deren Betreuer zu finden.

Linux Coding Style

- Quelltext in Linux werden *normalisiert* bearbeitet
- <http://lxr.linux.no/linux/Documentation/CodingStyle>
- Im Einzelnen: Einrückung, Lange Zeilen, Setzen von Klammern und nichtdruckbare Zeichen (Whitespace), Bezeichnernamen, Tabulatoren, Kconfig, etc.

- ⇒ Erleichtert das Lesen und Verständnis, vermeidet “Streit”
- ⇒ Verstöße gegen die Richtlinien führen häufig zur Ablehnung des Patches (oder schlimmeres).

Entwicklungen absegnen

- In der Linux-Entwicklung wird großer Wert auf korrekte Zuordnung von Patches zu Maintainern gelegt
- Bei einzelnen, kleineren Patches ist häufig unklar, wer der eigentliche Autor ist und was die genauen Nutzungsbedingungen sind
- Per Konvention werden daher Patches von ihren jeweiligen Autoren mit einer speziellen Notation in der Commit-Nachricht *abgesegnet*:

```
Signed-off-by: Random J Developer <random@developer.example.org>
```

Bedeutung (Auszug, Details in Documentation/SubmittingPatches):

- Der Autor bestätigt das Werk ganz oder in Teilen selbst geschrieben zu haben.
- Der Autor bestätigt das Recht zur Veröffentlichung zu haben
- Der Autor erlaubt die Verwendung und den Vertrieb der Änderung unter den Bedingungen der ursprünglichen Version

Weitere Absegnungen

- **Reported-by:** Wer hat das Problem (richtig) gemeldet
- **Reviewed-by:** Wer hat die Änderung begutachtet
- **Tested-by:** Wer hat den Patch getestet
- **Acked-by:** Patch ist zur Kenntnis genommen worden, nicht notwendigerweise aber getestet

Übungsaufgabe #3

- Programmcode des Linux-Staging-Bereich des aktuellen Staging-Baums analysieren und Defekte finden
- Patches für gefundene Fehler erstellen und an `linux-kernel@i4.informatik.uni-erlangen.de` senden
- Vorstellung der Ergebnisse in der Tafelübung (Kurzvortrag, maximal drei Folien pro Gruppe)
- Einsendung der Patches an `devel@linuxdriverproject.org` mit Kopie (CC:) an die zuständigen Betreuer

Abgabe: 20.06.2011